LOCKSTEP CLIENT-SERVER ARCHITECTURE

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] This application claims the benefit of U.S. Provisional Patent Application No. 62/811,120, filed Feb. 27, 2019, the entire contents of which are incorporated by reference herein.

BACKGROUND

[0002] Client applications, particularly games, often send and receive instructions from a centralized server. For example, in a gaming application, the player can request that a server perform a particular action on each client device currently active in the game. Communication between the client devices and centralized server is critical to maintain efficient gameplay. Due to the centralized nature of the game architecture, the game would go offline when the server goes offline of otherwise fails. This may result in interrupted gameplay.

DESCRIPTION OF THE DRAWINGS

[0003] FIG. **1** is a schematic diagram of a first example lockstep client-server system, in accordance with embodiments of the disclosure.

[0004] FIG. **2** is a schematic diagram of a second example lockstep client-server system, in accordance with embodiments of the disclosure.

[0005] FIG. **3** is an illustration of an example of a method of lockstep client-server operations, in accordance with embodiments of the disclosure.

[0006] FIG. **4** is an illustration of an example of a method of lockstep client-server replay operations, in accordance with embodiments of the disclosure.

[0007] FIG. **5** is an illustration of an example of a method of lockstep client-server recommendation operations, in accordance with embodiments of the disclosure.

[0008] FIG. **6** is a block diagram of an example computing device, in accordance with embodiments of the disclosure.

DETAILED DESCRIPTION

[0009] Aspects of the disclosure relate to lockstep client-server architecture. Specifically, the present disclosure is directed to systems and methods for running a real-time multiplayer game (e.g., a mobile game, such as a massively multiplayer online ("MMO") game or the like) or other suitable client application with a server and multiple clients in synchronized agreement about the state of the game or other client application.

[0010] For example, a use case for the present invention may include a client application ("app") that can host real-time matches with multiple participants. Each person may play on their own client device (e.g., a mobile device, such as a smartphone, a tablet computer, or the like) that is connected to a server.

[0011] Merely for purposes of illustration and not limitation, a tower-defense game that contains multiplayer battle matches will be used as an example of such a client application to assist in describing the present invention. According to the present illustration, each instance of a match has a start state, and begins synchronized among the players. A match has a finite duration. Players are free to choose what inputs (e.g., actions, tasks, etc.) to make and when to make them, within the rules of the game. The behavior of the match plays out over time, affected by the inputs made by the players in real-time. It is expected that all client devices experience the same match, with the same sequence of inputs and the same outcome.

[0012] In one embodiment, gameplay may become interrupted if a centralized server controlling the gameplay goes offline or otherwise experiences an error. Advantageously, the methods and systems disclosed herein overcome the above challenges, and others, by maintaining synchronized applications on each of the client devices and server(s).

[0013] In one embodiment, processing logic of a computer processing device may send, by a server to each of a plurality of client devices, initializing information at a start of a task in a client application running on each of the plurality of client devices. Initializing information may include setup data that allows each client device and server to execute identical copies of deterministic applications such that the same actions may be performed by each of the client devices and servers at the same time without communication with each other.

[0014] Processing logic may further receive, by the server, an input message from one of the plurality of client devices, wherein the input message is generated from an interaction with the task in the client application. In one embodiment, the input message may include a request from a user to perform the particular task (e.g., a move in a game, an annotation to a shared document, etc.), as well as an identifier of the frame (e.g., a graphics or video frame) during which the task should be performed.

[0015] Processing logic may further generate, by a computer processing device of the server, updated information for the task in the client application based on the received input message. For example, in one embodiment, the updated information may include instructions to perform the particular task at a particular time (e.g., as indicated by a frame identifier). In one embodiment, the particular time may be determined based on a requested time or a time generated by the server.

[0016] Processing logic may further send, by the server, the updated information to each of the plurality of client devices while the task in the client application is running to maintain synchronism between the server and the plurality of client devices for the task. Processing logic may further record a log of each of the actions performed in a particular task of a client application, such that tasks of applications may be replayed without replaying a video of the tasks being performed. In other words, games may be replayed and displayed by re-executing all actions taken during the game, without recording the actual gameplay itself. Advantageously, such techniques may result in a much smaller data file being maintained or otherwise substantially reduced data storage requirements (e.g., a video/screen capture is not required to be stored) while allowing all replay features to be performed (e.g., fast forward, rewind, pause, etc.).

[0017] Although examples of the disclosure may be described in the context of a mobile video game application, such examples are for illustrative purposes only. Aspects of the disclosure may be utilized by any client applications that use any kind of collaborative environment and/or entities. For example, aspects of the disclosure may be used by collaborative editing software, collaborative presentation software, collaborative viewing software, etc.